Table 1 (cont')

```
/* do the alignment, return best score: main()
 * dna: values in Fitch and Smith, PNAS, 80, 1382-1386, 1983
 * pro: PAM 250 values
 * When scores are equal, we prefer mismatches to any gap, prefer
 * a new gap to extending an ongoing gap, and prefer a gap in seqx
 * to a gap in seq y.
 */
nw()                                                                    nw
{
        char            *px, *py;               /* seqs and ptrs */
        int             *ndely, *dely;  /* keep track of dely */
        int             ndelx, delx;    /* keep track of delx */
        int             *tmp;           /* for swapping row0, row1 */
        int             mis;            /* score for each type */
        int             ins0, ins1;     /* insertion penalties */
        register        id;             /* diagonal index */
        register        ij;             /* jmp index */
        register        *col0, *col1;   /* score for curr, last row */
        register        xx, yy;         /* index into seqs */

        dx = (struct diag *)g_calloc("to get diags", len0+len1+1, sizeof(struct diag));

        ndely = (int *)g_calloc("to get ndely", len1+1, sizeof(int));
        dely = (int *)g_calloc("to get dely", len1+1, sizeof(int));
        col0 = (int *)g_calloc("to get col0", len1+1, sizeof(int));
        col1 = (int *)g_calloc("to get col1", len1+1, sizeof(int));
        ins0 = (dna)? DINS0 : PINS0;
        ins1 = (dna)? DINS1 : PINS1;

        smax = -10000;
        if (endgaps) {
                for (col0[0] = dely[0] = -ins0, yy = 1; yy <= len1; yy++) {
                        col0[yy] = dely[yy] = col0[yy-1] - ins1;
                        ndely[yy] = yy;
                }
                col0[0] = 0;            /* Waterman Bull Math Biol 84 */
        }
        else
                for (yy = 1; yy <= len1; yy++)
                        dely[yy] = -ins0;

        /* fill in match matrix
         */
        for (px = seqx[0], xx = 1; xx <= len0; px++, xx++) {
                /* initialize first entry in col
                 */
                if (endgaps) {
                        if (xx == 1)
                                col1[0] = delx = -(ins0+ins1);
                        else
                                col1[0] = delx = col0[0] - ins1;
                        ndelx = xx;
                }
                else {
                        col1[0] = 0;
                        delx = -ins0;
                        ndelx = 0;
                }
```

Table 1 (cont')


```
for (py = seqx[1], yy = 1; yy <= len1; py++, yy++) {
        mis = col0[yy-1];
        if (dna)
                mis += (xbm[*px-'A']&xbm[*py-'A'])? DMAT : DMIS;
        else
                mis += _day[*px-'A'][*py-'A'];

        /* update penalty for del in x seq;
         * favor new del over ongong del
         * ignore MAXGAP if weighting endgaps
         */
        if (endgaps || ndely[yy] < MAXGAP) {
                if (col0[yy] - ins0 >= dely[yy]) {
                        dely[yy] = col0[yy] - (ins0+ins1);
                        ndely[yy] = 1;
                } else {
                        dely[yy] -= ins1;
                        ndely[yy]++;
                }
        } else {
                if (col0[yy] - (ins0+ins1) >= dely[yy]) {
                        dely[yy] = col0[yy] - (ins0+ins1);
                        ndely[yy] = 1;
                } else
                        ndely[yy]++;
        }

        /* update penalty for del in y seq;
         * favor new del over ongong del
         */
        if (endgaps || ndelx < MAXGAP) {
                if (col1[yy-1] - ins0 >= delx) {
                        delx = col1[yy-1] - (ins0+ins1);
                        ndelx = 1;
                } else {
                        delx -= ins1;
                        ndelx++;
                }
        } else {
                if (col1[yy-1] - (ins0+ins1) >= delx) {
                        delx = col1[yy-1] - (ins0+ins1);
                        ndelx = 1;
                } else
                        ndelx++;
        }

        /* pick the maximum score; we're favoring
         * mis over any del and delx over dely
         */
```

Table 1 (cont')


```
                id = xx - yy + len1 - 1;
                if (mis >= delx && mis >= dely[yy])
                        col1[yy] = mis;
                else if (delx >= dely[yy]) {
                        col1[yy] = delx;
                        ij = dx[id].ijmp;
                        if (dx[id].jp.n[0] && (!dna || (ndelx >= MAXJMP
                        && xx > dx[id].jp.x[ij]+MX) || mis > dx[id].score+DINS0)) {
                                dx[id].ijmp++;
                                if (++ij >= MAXJMP) {
                                        writejmps(id);
                                        ij = dx[id].ijmp = 0;
                                        dx[id].offset = offset;
                                        offset += sizeof(struct jmp) + sizeof(offset);
                                }
                        }
                        dx[id].jp.n[ij] = ndelx;
                        dx[id].jp.x[ij] = xx;
                        dx[id].score = delx;
                }
                else {
                        col1[yy] = dely[yy];
                        ij = dx[id].ijmp;
        if (dx[id].jp.n[0] && (!dna || (ndely[yy] >= MAXJMP
                        && xx > dx[id].jp.x[ij]+MX) || mis > dx[id].score+DINS0)) {
                                dx[id].ijmp++;
                                if (++ij >= MAXJMP) {
                                        writejmps(id);
                                        ij = dx[id].ijmp = 0;
                                        dx[id].offset = offset;
                                        offset += sizeof(struct jmp) + sizeof(offset);
                                }
                        }
                        dx[id].jp.n[ij] = -ndely[yy];
                        dx[id].jp.x[ij] = xx;
                        dx[id].score = dely[yy];
                }
                if (xx == len0 && yy < len1) {
                        /* last col
                         */
                        if (endgaps)
                                col1[yy] -= ins0+ins1*(len1-yy);
                        if (col1[yy] > smax) {
                                smax = col1[yy];
                                dmax = id;
                        }
                }
        }
        if (endgaps && xx < len0)
                col1[yy-1] -= ins0+ins1*(len0-xx);
        if (col1[yy-1] > smax) {
                smax = col1[yy-1];
                dmax = id;
        }
        tmp = col0; col0 = col1; col1 = tmp;
}
(void) free((char *)ndely);
(void) free((char *)dely);
(void) free((char *)col0);
(void) free((char *)col1);
}
```